Research Paper | Computer Science | E-ISSN No : 2454-9916 | Volume : 3 | Issue : 5 | May 2017

NEXT WORD PREDICTION AND CORRECTION SYSTEM USING TENSORFLOW

Prasad R

Dept. of ISE, BMS college of Engineering, Bangalore, India – 560 019.

ABSTRACT

As per the title, the paper presents the concept on language processing. Natural language processing is a field of science and engineering where humans and the computers are interacted. With respect to the computer system, An Artificial Intelligence is also a field of science and technology where the computer system should act like a human intelligence.

Now a days, peoples are giving their feedback/reviews/ comments in social media or other medium with shortcuts and spelling mistakes. So goal is to predict that misspelted word and correct it with word to vector and recurrent neural networks models using tensorflow.

KEYWORDS: Next/Target word, Phrases, Probability, Perplexity, Performance.

INTRODUCTION:
The Introduction presents the purpose of the studies and relationship to previous work in the field. It is not required to incorporate an extensive review of the literature. Use only recent references and provide the most salient information to allow the readers to understand and evaluate the purpose and results of the present study.

Components of NLP are :
1. Natural Language Understanding (NLU)
- Mapping of given input into useful representation using natural language.
- Analysis of different perspectives of the natural language.

2. Natural Language Generation (NLG)
- **Text planning:** This includes receiving the relevant content from learned knowledge base.
- **Sentence planning:** This includes selecting required words and forming meaningful phrases and setting tone for the sentence.

Tensor is a central unit of data in TensorFlow. A tensor in a tensorflow consists of a set of values shaped into an array of different number of dimensions. A Rank of tensor's is the number of dimensions which the tensor has.

3 – this is a rank 0 tensor with scalar and shape []

[1. ,2., 3.] – this is a rank 1 tensor and vector with shape [3]

[[1., 4., 5.], [5., 8., 9.]] – this is a rank 2 tensor and the matrix with shape [2, 3]

[[[1., 8., 9.]], [[3., 6., 9.]]] – this a rank 3 tensor with shape is [2, 1, 3]

Basic working of tensorflow is explained below
1. Importing Tensorflow:
Import tensorflow as tf
This statement gives python access to all of Tensorflow's classes, methods, and symbols.

2. Graph computation:
- Building the computational graph.

This graphs can be build and design by a series of tensorflow operations and henerated as a graph of nodes.

- Running the computational graph.

To evaluate the nodes, the computation graph should be run within a tensorflow session. This session encapsulates the control and state of the tensorflow runtime.

3. Building softmax regressions:
softmax regression is a technique of assigning probabilities to each objects by differentiating similarity in that graph.

4. Training the model:
In training the model, train, valid and test datasets are created for prediction of words. Train datasets are used to primary training, validation datasets are used for check the validation of training accuracy and test datasets are used for final testing of the accuracy.

5. Evaluate the model:
Evaluating of model is used to evaluate the trained model to check whether it gives better results or not.

MATERIALS AND METHODS:
System model:
The different methodologies can give better prediction and can be categorized into two namely count-based methods, Ex. Latent Semantic Analysis and predictive methods, Ex. neural probabilistic language models. Count-based methods compute the prediction using statistics of the trained system and prediction is based on how often some word co-occurs with its neighbour words in a large corpus of text and then mapping of these count statistics down to a small, dense vector for each word. Predictive models are directly predict a word from its neighbour words in terms of learned small, dense embedding vectors [2].

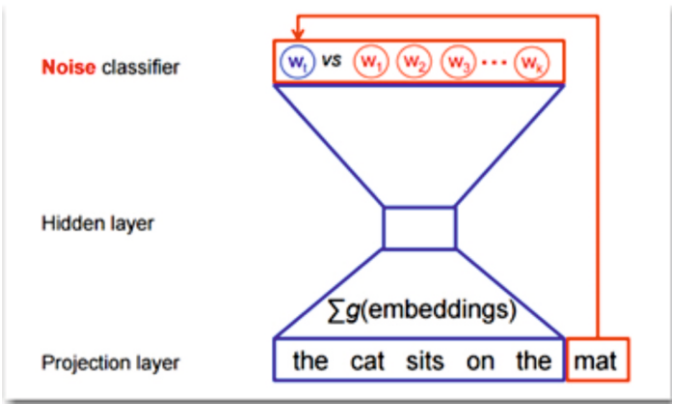


Fig. 2.1 CBOW and skip gram model

In the Fig.2.1, Continuous Bag Of Words (CBOW) predicts next or target word. Consider a sentence "'the cat sits on the", 'mat' from source context words "'the cat sits on the", and the skip-gram does the reverse operation of CBOW and predicts source context words from the next or target words. This inversion looks like an arbitrary and random choice, but statistically it has the effect that CBOW smoothes over a lot of the distributional information. But it is more useful for smaller datasets and gives better prediction. However, skip gram treats each context-target pair as a new observation and this model leads to do better prediction when the datasets requires larger. [2]

The skip gram model forms a moderate to larger dataset of words and defines the contexts in different situations. For example, it defines words to the left of the next or target, words to the right of the next or target.

Consider a sentence "this phone battery is better than camera" with skip window size 1. The following are the context and target word pairs(context, target).

([this, battery], phone), ([phone, battery], is), ([battery, better], is), ... so on

Skip gram provides a prediction for each context word from its target word then datasets becomes are in the form of (input, output)

(phone, this), (phone, battery), (is, phone), (is, battery), ... so on.

Recurrent neural networks (RNN) is a language model used to find and assigns probabilities to sentences by predicting next or target words in a text from the history of previous words. This model uses the concept the Penn Tree Bank (PTB) dataset, where it is a most popular benchmark for finding the quality of sentences and it is small and fast to train. The LSTM core model contains LSTM cell that processes one word at a time and assigns probabilities of the most possible values for the next or target word in the sentence.

In Recurrent neural networks (RNN), the word "recurrent" means "persistent". The network having indefinite continuous neural networks is called Recurrent neural networks. Recurrent neural networks addressed this above issue. These RNNs allows persistent information with loops in them and are shown in the below figure.

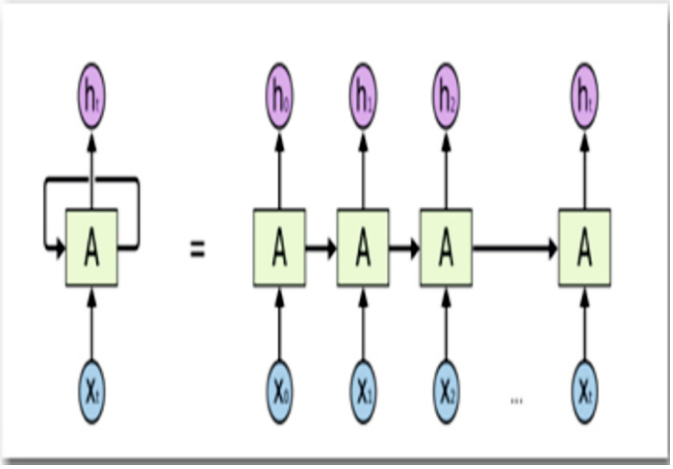


Fig. 2.2 Recurrent neural networks model with loops

In the figure-2.2, consider a chunk of neural network, 'A', has some input Xt and outputs a value ht. Information of loops passed from one neural network to other neural network. A recurrent neural network (RNN) can be considered of as multiple copies of the same neural network, each network passing a message to a successor. RNNs is used to solve different kinds of problems such as speech recognition, language modelling, translation, image capturing[1].

RNN can connect past data or information to the present information but it gives some problem in long term dependencies. For example, consider a sentence "clouds are in the sky", to predict the last word in this sentence is easy because the gap between target word from the previous words information is small and there is no context in the sentence but consider an other sentence. "I grew up in India… I speak fluent Hindi". To predict the last word "Hindi" is depends not only on previous word information but also on the context and the gap between present information and past information is more. To overcome this problem, LSTM (Long Short Term Memory) networks are used[1].

Proposed Methodology:
The proposed design consists of four blocks namely Database, Computer system, Tensorflow model, Training System and it is showing in the figure 2.3. In our database is having thousands of sentences need to process at a faster rate so that the performance of the system should increase. Second block is computer system where the system should support the requirements for the tensorflow model to communicate. Training model invokes tensorflow whenever the it calls by the computer system.

Main goal of this proposed design is to achieve misspelt prediction and correction in a sentence.

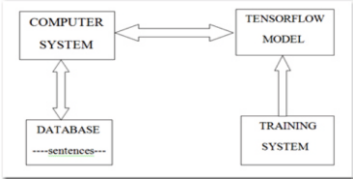


Fig. 2.3 Architecture of working model

RESULTS:
The experimental result for the sentence "this phone battery is better thn camera" is shown in the figure 3.1.

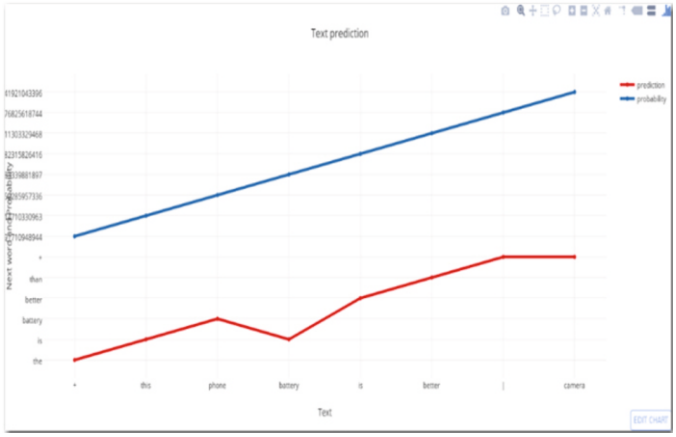
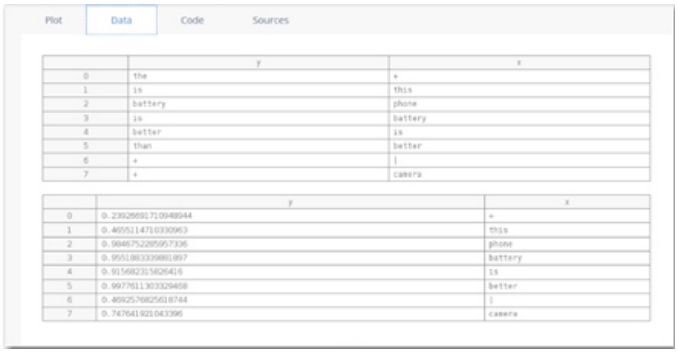


Fig. 3.2 Graph of Next Word Prediction With Probability



	y	x
0	the	+
1	is	this
2	battery	phone
3	is	battery
4	better	is
5	than	better
6	+		
7	+	camera

	y	x
0	0. 23926991710948944	+
1	0. 4655114710330963	this
2	0. 9846752285957336	phone
3	0. 9551883339881897	battery
4	0. 9156823515626416	is
5	0. 9977611303329468	better
6	0. 4692576825618744		
7	0. 7476419210433996	camera

Fig. 3.3 Data of Next Word Prediction With Probabilities

Performance:
The performance of the system can be measured by calculating number of sentences it can process in a short period of time (seconds).

DISCUSSION:
In social media, the reviews, comments and tweets are all having a words which are misspelted and written in shortcuts. This can be easily understand by a human beings because of the daily routines towards social medias are common to humans but system should not understand these misspellings and shortcuts. So system requires training to understand thse kind of reviews.

Following are implementations involved in this paper

1. Next word prediction using target word.
Next word prediction is one of the most important task that the system should do. Consider a sentence "this OS is gud". This sentence has both shortcut word "OS" and misspelted word "gud". By using tensorflow, the system will recognize next words for "OS" as "Operating System" and "gud" as "good"

2. Phrases prediction.
In some sentences for example, "iphone camera is better then other phones" the word "then" is wrong in that case even though the word "then" is not a misspelted word but the sentence gives the comparison between phones so it should be a word "than". So this phrase prediction based on context is more important.

3. Spelling prediction and correction system.
Prediction of misspelted word can be found by considering misspelted word as a unknown token and correcting it from next word prediction mechanism. Consideration of word as unknown token is depending on the "numpy" files. Numpy files are the tensorflow library where it provides numerical computations. This system can create a numpy files for all the trained corpus text and these trained sentences are saved in the numpy files. Also, Penn tree bank (PTB) datasets are created using H5(HDF5) files. These HDF files are hierarchical data format files where it creates hierarchical tree structure for each sentences and saved into the h5 file. This h5 file helps in prediction of next or target words in different contexts with better results.

4. Closed loop tensorflow system.
Closed loop system in tensorflow is used to improve the prediction level of a words. In this mechanism, the some trained and untrained sentences are trained again by looping them back to the tensorflow training system. For example, suppose system is trained with 10000 sentences and if we need to train a system with 100 more sentences then tensorflow closed loop mechanism is more useful.

CONCLUSIONS:

This paper presents how the system is predicting and correcting the next/target words using some mechanisms and using tensorflow closed loop system, the scalability of trained system can be increased and using perplexity concept the system will decide that the sentence is having more misspelts and the performance of the system can be increased.

This product has more scope on social media for syntax analysis and semantic analysis in natural language processing in Artificial intelligence.

REFERENCES:

1. Wojciech Zaremba, Ilya Sutskever, Oriol Vinyals, Recurrent neural network regularization.
2. Tomas mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean, Distributed representations of words and phrases.
3. Yoshua Bengio; Rejean Ducharme and Pascal Vincent, A neural probabilistic model.
4. Yoshua Bengio, R´ejean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. The Journal of Machine Learning Research, 3:1137–1155, 2003.
5. D. Baker and A. McCallum. Distributional clustering of words for text classification. In SIGlR '98,1998.
6. R. Miikkulainen and M.G. Dyer. Natural language processing with modular neural networksand distributed lexicon. Cognitive Science, 15:343- 399, 1991.
7. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. ICLR Workshop, 2013